Probabilistic Temporal Logic for Reasoning about Bounded Policies

Nima Motamed[1] , **Natasha Alechina**[1] , **Mehdi Dastani**[1] , **Dragan Doder**[1] and **Brian Logan**[1,2]
[1]Utrecht University
[2]University of Aberdeen
{n.motamed, n.a.alechina, m.m.dastani, d.doder, b.s.logan}@uu.nl

Abstract

To build a theory of intention revision for agents operating in stochastic environments, we need a logic in which we can explicitly reason about their decision-making policies and those policies' uncertain outcomes. Toward this end, we propose PLBP, a novel probabilistic temporal logic for Markov Decision Processes that allows us to reason about policies of bounded size. The logic is designed so that its expressive power is sufficient for the intended applications, whilst at the same time possessing strong computational properties. We prove that the satisfiability problem for our logic is decidable, and that its model checking problem is PSPACE-complete. This allows us to e.g. algorithmically verify whether an agent's intentions are coherent, or whether a specific policy satisfies safety and/or liveness properties.

1 Introduction

To design and develop intelligent and autonomous agents, it is essential to build a framework describing when and how they act. Such a framework should allow agents to commit to certain actions and to revise their commitments later on. Creating such a framework, therefore, requires the study of intention revision, a topic receiving increased interest in recent decades [Cohen and Levesque, 1990; Rao and Georgeff, 1991; van der Hoek *et al.*, 2007; Shoham, 2009; Icard *et al.*, 2010; van Ditmarsch *et al.*, 2011; van Zee *et al.*, 2020].

As observed by [Shoham, 2009], a primitive yet important kind of intention is that of commitment to perform an action at a specific time. This already comes with non-trivial complications, involving temporal reasoning about the pre- and postconditions of actions. To build a formal framework capable of dealing with this notion of intention, a natural approach is to start with an appropriate underlying temporal logic, as done by [Icard *et al.*, 2010; van Zee *et al.*, 2020]. In their work, actions are deterministic, with a single outcome.

For many practical applications, agents are expected to operate using behavioral policies in stochastic environments where actions have uncertain, nondeterministic outcomes. So in order to study intention revision using methods similar to those of [Icard *et al.*, 2010; van Zee *et al.*, 2020], we require an appropriate probabilistic temporal logic on which to build the model. Such a logic should allow us to reason about the execution, precondition, and (possibly many) postconditions of actions and policies. And in order to be useful in practice, the logic should possess strong computational properties such as efficient (or at least, decidable) model checking and satisfiability, so that we can e.g. algorithmically verify properties of policies and compute whether certain logical inferences are valid.

Probabilistic temporal logics in wide use include PCTL [Hansson and Jonsson, 1994], pCTL* [Aziz *et al.*, 1995], PATL/PATL* [Chen and Lu, 2007], and Probabilistic Strategy Logic [Aminof *et al.*, 2019]. None of these logics fit our desiderata. First, they do not allow us to reason about pre- and postconditions of specific actions. Second, they generally have either high complexity or even undecidable model checking. Note that the decidability of the satisfiability problem for any of these logics would imply that of PCTL, which is a known open problem, and so we cannot get decidable satisfiability for these logics without severe restrictions. Third, from a more practical point of view, all of these logics are interpreted over infinite traces, while finite traces are both sufficient and more natural for most applications in AI (see e.g. [Artale *et al.*, 2019]).

The contribution of this paper is the introduction of the Probabilistic Logic of Bounded Policies (PLBP), a novel probabilistic temporal logic fitting our desiderata, interpreted over finite traces and bounded-time policies in Markov Decision Processes. The logic allows for both reasoning about specific actions/policies, as well as the existence of policies satisfying certain properties. We prove that the model checking problem for PLBP is PSPACE-complete, that it possesses the small model property, and that the satisfiability problem is decidable with a 2EXPSPACE algorithm. The novelty of the logic lies in the fact that it enables us to express a host of properties important for both general applications in AI, as well as for our intended applications, while simultaneously maintaining strong computational properties, which is uncommon amongst probabilistic temporal logics.

2 Syntax and Semantics of PLBP

PLBP is defined relative to a countably infinite set Prop of *propositional variables* and a finite set $\mathscr{A}$ of *actions*. To each action $a \in \mathscr{A}$ we associate a *precondition* pre_a, which is a conjunction of literals over Prop, and a finite nonempty list Post_a of possible *postconditions*, which are also conjunctions of literals. Intuitively, the precondition of an action is precisely what must be satisfied so that the action can be executed, and a postcondition of an action is one of its possible outcomes, i.e. which is made true after executing the action. We refer to the ith postcondition of a as $\mathsf{post}_{a,i}$. We require postconditions of an action a to be mutually inconsistent, i.e. $\mathsf{post}_{a,i}$ and $\mathsf{post}_{a,j}$ cannot be simultaneously true for all $i \neq j$, meaning that there is a literal in $\mathsf{post}_{a,i}$ for which its negation is in $\mathsf{post}_{a,j}$. We refer to actions with a single postcondition as deterministic.

Example 1 (Running example)**.** *Mary is a final-year student in Artificial Intelligence. In order to graduate, she needs to pass her exam on methods in AI research. She's not completely sure yet what she wants to do: she can try to graduate and then apply for a Ph.D. position or for a position in industry, but part of her is also considering the possibility of going straight for a position in industry.*

There are several things she can do. She can try to study hard for the exam, but she can also take it easy and try the exam unprepared. If she fails the exam, she can try again next year. If she passes, she can then apply for a Ph.D. position or a position in industry, She can also just apply straight for a position in industry.

We formalize Mary's story through propositional variables pass, inPhD *and* $\mathsf{inIndustry}$. *The actions are* study, $\mathsf{takeEasy}$, $\mathsf{applyIndustry}$ *and* $\mathsf{applyPhD}$. *The preconditions are*
- $\mathsf{pre}_{\mathsf{study}} = \neg\mathsf{pass} \wedge \neg\mathsf{inIndustry} \wedge \neg\mathsf{inPhD}$,
- $\mathsf{pre}_{\mathsf{takeEasy}} = \neg\mathsf{pass} \wedge \neg\mathsf{inIndustry} \wedge \neg\mathsf{inPhD}$,
- $\mathsf{pre}_{\mathsf{applyIndustry}} = \neg\mathsf{inIndustry} \wedge \neg\mathsf{inPhD}$*, and*
- $\mathsf{pre}_{\mathsf{applyPhD}} = \mathsf{pass} \wedge \neg\mathsf{inIndustry} \wedge \neg\mathsf{inPhD}$.

The postconditions are
- $\mathsf{Post}_{\mathsf{study}} = \langle \mathsf{pass}, \neg\mathsf{pass} \rangle$,
- $\mathsf{Post}_{\mathsf{takeEasy}} = \langle \mathsf{pass}, \neg\mathsf{pass} \rangle$,
- $\mathsf{Post}_{\mathsf{applyIndustry}} = \langle \mathsf{inIndustry}, \neg\mathsf{inIndustry} \rangle$*, and*
- $\mathsf{Post}_{\mathsf{applyPhD}} = \langle \mathsf{inPhD}, \neg\mathsf{inPhD} \rangle$.

We will write $f : X \rightharpoonup Y$ to denote that f is a *partial* function from X to Y, and write $f(x)\downarrow$ to denote that f is defined on input $x \in X$. Given a set X, some $x \in X$ and $n \geqslant 1$, we write X_x^n for the set of all sequences in X of length n starting with x, and we write $X_x^{\leqslant n} = \bigcup_{k=1}^{n} X_x^k$. Finally, we write $\Delta(X)$ for the set of all finitely supported (discrete) probability distributions on X, i.e. those probability distributions $D : X \to [0, 1]$ such that $D(x) > 0$ for only finitely many x.

Definition 1 (MDP)**.** *A* Markov decision process *(MDP) over* $\mathscr{A}$ *is a tuple* $\mathbb{M} = \langle S, P, V \rangle$, *where* S *is the set of* states, $P : S \times \mathscr{A} \rightharpoonup \Delta(S)$ *is the* partial probabilistic transition function, *and* $V : S \to 2^{\mathsf{Prop}}$ *is the* valuation. *We often abbreviate* $P(s, a)$ *by* $P_{s,a}$. *These are required to satisfy the following conditions.*

(i) For all $s \in S$*, there is some* $a \in \mathscr{A}$ *such that* $s \models \mathsf{pre}_a$[1] *(or by the following condition, equivalently* $P_{s,a}\downarrow$*).*

(ii) $P_{s,a}\downarrow$ *iff* $s \models \mathsf{pre}_a$.

(iii) Given $P_{s,a}\downarrow$,

(iii.i) for all $t \in S$ *such that* $P_{s,a}(t) > 0$*, there is a unique* $\mathsf{post}_{a,i}$ *in* Post_a *such that* $t \models \mathsf{post}_{a,i}$*, and*

(iii.ii) for all $\mathsf{post}_{a,i}$ *in* Post_a *there is a unique* t *such that* $P_{s,a}(t) > 0$ *and* $t \models \mathsf{post}_{a,i}$.

Condition (i) states that MDPs have no deadlocks, which we include as it allows us to present the following definitions in a cleaner way. In our examples we will consider MDPs that technically do not satisfy this condition: this is not a problem, since we can add in a special deterministic 'do nothing' action which can be executed at any deadlocking state. Conditions (ii) and (iii) ensure that pre- and postconditions are meaningful: an action is executable precisely when the precondition holds, and the possible outcomes of an action are precisely the postconditions.

Example 2 (Running example, continued)**.** *An example MDP describing Mary's story using the actions given in Example 1 is* $\mathbb{M} = \langle S, P, V \rangle$*, where* $S = \{s_{\mathsf{student}}, s_{\mathsf{pass}}, s_{\mathsf{industry}}, s_{\mathsf{PhD}}\}$*, respectively denoting the states in which Mary is still a student, in which she passed the exam (and therefore graduated), in which she got a position in industry, and finally, in which she got a Ph.D. position. The valuation* V *is given by* $V(s_{\mathsf{student}}) = \emptyset$*,* $V(s_{\mathsf{pass}}) = \{\mathsf{pass}\}$*,* $V(s_{\mathsf{industry}}) = \{\mathsf{inIndustry}\}$*, and* $V(s_{\mathsf{PhD}}) = \{\mathsf{inPhD}\}$. *The partial probabilistic transition function* P *is defined (for all executable actions according to the preconditions) as*
- $P(s_{\mathsf{student}}, \mathsf{study})(s_{\mathsf{student}}) = 0.2$*, and* $P(s_{\mathsf{student}}, \mathsf{study})(s_{\mathsf{pass}}) = 0.8$ *(Mary is likely to pass when studying),*
- $P(s_{\mathsf{student}}, \mathsf{takeEasy})(s_{\mathsf{student}}) = 0.7$*, and* $P(s_{\mathsf{student}}, \mathsf{takeEasy})(s_{\mathsf{pass}}) = 0.3$ *(Mary is likely to fail when taking it easy),*
- $P(s_{\mathsf{student}}, \mathsf{applyIndustry})(s_{\mathsf{student}}) = 0.8$*,* $P(s_{\mathsf{student}}, \mathsf{applyIndustry})(s_{\mathsf{industry}}) = 0.2$ *(Mary is unlikely to just get into industry),*
- $P(s_{\mathsf{pass}}, \mathsf{applyIndustry})(s_{\mathsf{pass}}) = 0.4$*,* $P(s_{\mathsf{pass}}, \mathsf{applyIndustry})(s_{\mathsf{industry}}) = 0.6$ *(with a degree Mary has reasonable odds of getting into industry),*
- $P(s_{\mathsf{pass}}, \mathsf{applyPhD})(s_{\mathsf{pass}}) = 0.1$*,* $P(s_{\mathsf{pass}}, \mathsf{applyPhD})(s_{\mathsf{PhD}}) = 0.9$ *(Mary will almost surely get into a Ph.D. if she applies).*

Our logic will be built around the notion of a *bounded policy*, telling the agent how to act for a specified number of timesteps in a memoryful manner. These allow us to reason about the probabilities of sequences of states up to a certain length. While such policies might seem restrictive compared to more standard notions of policy considered in the literature, it suffices for our intended applications, as we will comment on at the end of this section. And importantly, while bounded policies are still memoryful (in a bounded sense), the amount of n-step bounded policies from a certain state in a finite MDP will always be finite, in contrast to general memoryful policies. This is an important property that we will use in proving the decidability of the satisfiability problem in Section 4.

On a similar note, we also restrict ourselves to *deterministic* policies, as they are simple and sufficient for most applica-

[1]Here, $\models$ is a standard classical satisfaction relation, that is, $s \models x$ iff $x \in V(s)$ for $x \in \mathsf{Prop}$, $s \models \neg\varphi$ iff $s \not\models \varphi$, and finally, $s \models \varphi \wedge \psi$ iff both $s \models \varphi$ and $s \models \psi$.

tions (note e.g. the result by [Puterman, 1994] showing the guaranteed existence of optimal deterministic policies).

Definition 2 (Bounded policies)**.** *Given an MDP $\mathbb{M}$ and $n \geqslant 1$, an n-*step $\mathbb{M}$-policy *is a pair $\langle s, \pi\rangle$ where $s \in S$ is referred to as its* initial state *(and we say the policy is* from s*), and $\pi\colon S_s^{\leqslant n} \to \mathscr{A}$ is a function such that $s_k \models \mathsf{pre}_{\pi(s_1\cdots s_k)}$ (or equivalently $P(s_k, \pi(s_1 \cdots s_k))\downarrow$) for all $s_1 \cdots s_k \in S_s^{\leqslant n}$. A* bounded $\mathbb{M}$-policy *is an n-step $\mathbb{M}$-policy for some n.*

We drop the $\mathbb{M}$ when the MDP is clear from the context. We will also often identify the bounded policy $\langle s, \pi\rangle$ with the function π. Whenever we wish to make the initial state explicit, by slight abuse of notation, we write π^s instead of the function π.

For an n-step policy π^s, we define $\mathsf{Paths}(\pi^s)$ *as the set*

$$\{s_1a_1 \cdots s_na_ns_{n+1} \mid s_1 = s \;\&\; \forall i.\pi^s(s_1s_2\cdots s_i) = a_i\}.$$

Given an n-step policy π^s, its path distribution *is the probability distribution $\mu_\pi^{\mathbb{M},s}$ over* $\mathsf{Paths}(\pi^s)$ *defined as*

$$\mu_\pi^{\mathbb{M},s}(s_1a_1\cdots s_na_ns_{n+1}) = \prod_{i=1}^{n} P(s_i, a_i)(s_{i+1}).$$

This extends to sets of paths in the standard way, i.e. $\mu_\pi^{\mathbb{M},s}(X) = \sum_{\mathbf{s}\in X} \mu_\pi^{\mathbb{M},s}(\mathbf{s})$. Whenever it is clear from the context, we drop the $\mathbb{M}$ from the superscript.

Our policies are defined with respect to finite state histories. For practical purposes, it can be useful to consider policies that consider state-action histories as well, so that the agent can base the choice of the next action depending on which action it executed in the past. Although we do not expand upon this for simplicity of presentation, we note that all of our work extends to such policies as well.

Example 3 (Running example, continued)**.** *Suppose Mary realizes that she has enough time and funds to stay in university for an extra year, and therefore wants to take it easy this year, only starting to seriously study next year. If she manages to graduate this year, she will try to get into a Ph.D. (surprised by the ease of obtaining a degree).*

This can be seen as Mary adopting the 2-step policy π from s_{student}, defined as
- $\pi(s_{\mathsf{student}}) = \mathsf{takeEasy}$,
- $\pi(s_{\mathsf{student}}s_{\mathsf{student}}) = \mathsf{study}$, *and*
- $\pi(s_{\mathsf{student}}s_{\mathsf{pass}}) = \mathsf{applyPhD}$.

Note that we need not consider other state sequences than the ones specified, as it is impossible to actually encounter those with this policy.

The path distribution on $\mathsf{Paths}(\pi^{s_{\mathsf{student}}})$ *can then be verified to take the following values in its support:*
- $\mu_\pi^{s_{\mathsf{student}}}(s_{\mathsf{student}}\ \mathsf{takeEasy}\ s_{\mathsf{student}}\ \mathsf{study}\ s_{\mathsf{student}})$
  $= P_{s_{\mathsf{student}},\mathsf{takeEasy}}(s_{\mathsf{student}}) \cdot P_{s_{\mathsf{student}},\mathsf{study}}(s_{\mathsf{student}})$
  $= 0.7 \cdot 0.2 = 0.14$,
- $\mu_\pi^{s_{\mathsf{student}}}(s_{\mathsf{student}}\ \mathsf{takeEasy}\ s_{\mathsf{student}}\ \mathsf{study}\ s_{\mathsf{pass}})$
  $= P_{s_{\mathsf{student}},\mathsf{takeEasy}}(s_{\mathsf{student}}) \cdot P_{s_{\mathsf{student}},\mathsf{study}}(s_{\mathsf{pass}})$
  $= 0.7 \cdot 0.8 = 0.56$,
- $\mu_\pi^{s_{\mathsf{student}}}(s_{\mathsf{student}}\ \mathsf{takeEasy}\ s_{\mathsf{pass}}\ \mathsf{applyPhD}\ s_{\mathsf{pass}})$
  $= P_{s_{\mathsf{student}},\mathsf{takeEasy}}(s_{\mathsf{pass}}) \cdot P_{s_{\mathsf{pass}},\mathsf{applyPhD}}(s_{\mathsf{pass}})$
  $= 0.3 \cdot 0.1 = 0.03$, *and*
- $\mu_\pi^{s_{\mathsf{student}}}(s_{\mathsf{student}}\ \mathsf{takeEasy}\ s_{\mathsf{pass}}\ \mathsf{applyPhD}\ s_{\mathsf{PhD}})$
  $m = P_{s_{\mathsf{student}},\mathsf{takeEasy}}(s_{\mathsf{pass}}) \cdot P_{s_{\mathsf{pass}},\mathsf{applyPhD}}(s_{\mathsf{PhD}})$
  $= 0.3 \cdot 0.9 = 0.27.$

Using this path distribution, we can now determine the probability that Mary is not *in a Ph.D. in two timesteps when applying π by summing up the probabilities of paths ending in a state other than s_{PhD}, resulting in a probability of 0.73.*

Definition 3 (Syntax)**.** *The* language of PLBP *is inductively defined by the grammar*

$$\begin{aligned} \varphi &::= \bot \mid x \mid \varphi \wedge \varphi \mid \neg\varphi \mid \Diamond^n_{\bowtie r}\Phi^n, \\ \Phi^0 &::= \varphi, \\ \Phi^{k+1} &::= \varphi \mid \mathsf{do}_a \mid \Phi^{k+1} \wedge \Phi^{k+1} \mid \neg\Phi^{k+1} \mid \mathsf{X}\Phi^k, \end{aligned}$$

where $x \in \mathsf{Prop}$, $a \in \mathscr{A}$, $n \geqslant 1$, $r \in \mathbb{Q} \cap [0,1]$, and $\bowtie \in \{<, =, >\}$. Formulas φ are referred to as state formulas *(or just PLBP formulas), and formulas Φ^n are referred to as n-*step path formulas *(or more generally,* path formulas*).*

In the definition of state formulas, $\bot$ stands for falsity. The modal formula $\Diamond^n_{\bowtie r}\Phi$ involves existential quantification over policies and states that "the agent can act in the next n steps in such a way that Φ will hold with probability $\bowtie r$."

Boolean connectives $\vee, \to$ are defined as abbreviations in a standard way. We also define the following abbreviations:
- $\Diamond^n_{\geqslant r}\Phi := \Diamond^n_{>r}\Phi \vee \Diamond^n_{=r}\Phi$
- $\Diamond^n_{\leqslant r}\Phi := \Diamond^n_{<r}\Phi \vee \Diamond^n_{=r}\Phi$

We can also define a modal operator $\Box^n_{\bowtie r}\Phi$ that replaces the existential quantification with universal quantification, through the following abbreviations:
- $\Box^n_{<r}\Phi := \neg\Diamond^n_{>r}\Phi \wedge \neg\Diamond^n_{=r}\Phi$
- $\Box^n_{\leqslant r}\Phi := \neg\Diamond^n_{>r}\Phi$
- $\Box^n_{=r}\Phi := \neg\Diamond^n_{<r}\Phi \wedge \neg\Diamond^n_{>r}\Phi$
- $\Box^n_{\geqslant r}\Phi := \neg\Diamond^n_{<r}\Phi$
- $\Box^n_{>r}\Phi := \neg\Diamond^n_{<r}\Phi \wedge \neg\Diamond^n_{=r}\Phi$

Path formulas are built from state formulas together with do_a-propositions, combined using the X-operator and Boolean connectives. The do_a-proposition, where $a \in \mathscr{A}$, should be read as "the agent will now execute a", i.e. the next action on the path is a. The operator X stands for the next-time operator, with $\mathsf{X}\varphi$ being true on a path meaning that φ is true on the next state in the path.

Note that in the way we set up our syntax, Φ^n-formulas have at most n nested X-operators. Therefore, in the state formula $\Diamond^n_{\bowtie r}\Phi$, the path formula Φ is not allowed to have a nesting depth of X-operators greater than n. Intuitively, this is because the policies quantified over in $\Diamond^n_{\bowtie r}\Phi$ only look n steps ahead.[2]

If the limitation on the length of paths and nesting depth is dropped, problems arise in the semantics: we somehow

[2]The definition of path formulas could have been stated in a standard way, as

$$\varphi \mid \mathsf{do}_a \mid \Phi \wedge \Phi \mid \neg\Phi \mid \mathsf{X}\Phi,$$

but then in the definition of state formulas, we would need a side condition that in $\Diamond^n_{\bowtie r}\Phi$, the path formula Φ has at most n nested X-operators.

need to evaluate $\mathsf{X}\Phi$ in a state that does not exist on the path. In LTL on finite traces [De Giacomo and Vardi, 2015], all formulas $\mathsf{X}\Phi$ are false in the last state of a path, even if Φ is a tautology. So, paradoxically, the probability of a set of 1-step paths where a tautology holds after two steps would have to be 0 if we adopt a similar approach.

Note that although the temporal part of the logic only contains X, we can define abbreviations for bounded versions of other standard temporal operators like U (something holds until something else does) and G (something always holds).

E.g. $\mathsf{G}^n\Phi$, meaning Φ always holds in the next n steps, can be considered as an abbreviation of $\Phi \wedge \mathsf{X}\Phi \wedge \cdots \wedge \mathsf{X}^n\Phi$, where X^n is X repeated n times.

Example 4 (Running example, continued)**.** *Below are some examples of formulas of PLBP for the example of Mary:*

- inPhD *is a state formula and therefore also an n-step path formula for any n,*
- XinPhD *is an n-step path formula for $n \geqslant 1$, meaning that Mary is in a Ph.D. position at the next timestep,*
- $\lozenge^2_{=0.96}\mathsf{XXinPhD}$ *is a state formula, meaning that Mary can act now and in the next timestep in such a way that afterwards she will be in a Ph.D. position with probability 0.96, and*
- $\lozenge^2_{>0.4}\mathsf{XXXinPhD}$ *is* not *a syntactically correct formula because* $\mathsf{XXXinPhD}$ *has three nested* X*-operators, which cannot occur in the scope of a $\lozenge^2$ modality. Intuitively, it is not meaningful to say that Mary can act now and in the next timestep in such a way that in three steps she will be in a Ph.D. position with a probability greater than 0.4 since this would depend on what action Mary would take in two steps.*

Definition 4 (Semantics)**.** *Given an MDP $\mathbb{M}$, the* semantics of PLBP *is defined via simultaneous induction over state and path formulas: for state formulas, it is defined with respect to states as*

- $\mathbb{M}, s \models \bot$ *never,*
- $\mathbb{M}, s \models x$ *iff* $x \in V(s)$,
- $\mathbb{M}, s \models \varphi \wedge \psi$ *iff* $\mathbb{M}, s \models \varphi$ *and* $\mathbb{M}, s \models \psi$,
- $\mathbb{M}, s \models \neg\varphi$ *iff* $\mathbb{M}, s \not\models \varphi$, *and*
- $\mathbb{M}, s \models \lozenge^n_{\bowtie r}\Phi$ *iff there exists an n-step policy π^s such that* $\mu^s_\pi(\{\mathrm{s} \in \mathsf{Paths}(\pi^s) \mid \mathbb{M}, \mathrm{s} \models \Phi\}) \bowtie r$,

while for n-step path formulas, it is defined with respect to length n paths, i.e. nonempty finite sequences $\mathrm{s} = s_1a_1\cdots s_na_ns_{n+1}$ *of states and actions, as*

- $\mathbb{M}, \mathrm{s} \models \varphi$ *iff* $\mathbb{M}, s_1 \models \varphi$,
- $\mathbb{M}, \mathrm{s} \models \mathsf{do}_a$ *iff* $a_1 = a$,
- $\mathbb{M}, \mathrm{s} \models \Phi \wedge \Psi$ *iff* $\mathbb{M}, \mathrm{s} \models \Phi$ *and* $\mathbb{M}, \mathrm{s} \models \Psi$,
- $\mathbb{M}, \mathrm{s} \models \neg\Phi$ *iff* $\mathbb{M}, \mathrm{s} \not\models \Phi$, *and*
- $\mathbb{M}, \mathrm{s} \models \mathsf{X}\Phi$ *iff* $\mathbb{M}, \mathrm{s}_\mathsf{X} \models \Phi$ *where* $\mathrm{s}_\mathsf{X} = s_2a_2\cdots s_n$.

Whenever it is clear from the context, we drop the $\mathbb{M}$ and just write $s \models \varphi$ and $\mathrm{s} \models \Phi$.

Example 5 (Running example, continued)**.** *Revisiting the MDP we specified for Mary's situation, we see that the following formulas are satisfied.*

- $s_{\mathsf{student}} \models \lozenge^2_{>0.5}\mathsf{XXinIndustry}$*: Mary can act now and in the next step in such a way that with probability greater than 0.5, she will be in industry in two steps. Namely, with the policy saying to study now, and then (no matter the outcome) apply to industry afterwards.*
- $s_{\mathsf{student}} \models \mathsf{pre}_{\mathsf{study}} \wedge \square^1_{\geqslant 0.6}(\mathsf{do}_{\mathsf{study}} \rightarrow \mathsf{Xpass})$*: Mary can execute* study*, and doing so will cause* pass *to hold afterwards with a probability at least 0.6.*
- $s_{\mathsf{student}} \models \lozenge^1_{=1}\mathsf{X}\square^1_{=1}\mathsf{X}\neg\mathsf{inPhD}$*: Mary can act now in such a way that it becomes guaranteed that no matter how she acts afterwards, she will not be in a Ph.D. position. Namely, by choosing to apply to industry now.*

Since we have explicit postconditions in the logic, we can even formulate properties of specific policies Mary can choose. For example, consider the 2-step policy that says to take it easy now, apply to industry afterwards if she passes, and otherwise study. We can state in our logic that under this policy, Mary will be in industry in two steps with a probability less than 0.2. This can be done by way of the formula

$$\lozenge^2_{<0.1}(\mathsf{do}_{\mathsf{takeEasy}} \wedge (\mathsf{Xpost}_{\mathsf{takeEasy},1} \rightarrow \mathsf{Xdo}_{\mathsf{applyIndustry}}) \wedge (\mathsf{Xpost}_{\mathsf{takeEasy},2} \rightarrow \mathsf{Xdo}_{\mathsf{study}}) \wedge \mathsf{XXinIndustry}),$$

remembering that $\mathsf{post}_{\mathsf{takeEasy},1} = \mathsf{pass}$ *and* $\mathsf{post}_{\mathsf{takeEasy},2} = \neg\mathsf{pass}$*. And indeed, this formula holds at s_{student}. Generalizing beyond the example of Mary, we can similarly state that policies in other MDPs satisfy properties with some probability, e.g. safety and liveness properties.*

Example 6 (Coherence of intentions)**.** *Let us now revisit the theory of intention. Given a finite set I of "commitment towards time" intentions (which we take to be pairs $\langle a, t\rangle$ of actions a and time steps t, meaning that the agent intends to perform a at t), it is important to be able to determine whether adopting these intentions is coherent, in the sense laid out by [Shoham, 2009; van Zee* et al.*, 2020]. In particular, this means that intentions need to be internally consistent (e.g. at most one action can be executed at a time, and subsequent actions need to have compatible pre- and postconditions). Furthermore, with respect to the agent's beliefs, coherence means that for all of the agent's intentions, the agent does not believe that the preconditions of the intended action do not hold at the intended time, and the agent* does *believe that the intended actions' postconditions hold.*

Representing the agent's beliefs as a set Γ of formulas, we can formulate a notion of coherence of I with respect to Γ satisfying the above criteria. Consider the formula

$$\mathsf{exec}_\theta(I) = \lozenge^{t_{\max}}_{\geqslant\theta} \bigwedge_{\langle a,t\rangle \in I} \mathsf{X}^t\mathsf{do}_a,$$

where $t_{\max} = \max_{\langle a,t\rangle\in I} t$. This formula encodes the agent's belief (with strength $\theta \in [0, 1]$) that it is possible to execute all of their intentions. Using this, we can say that I is coherent with respect to Γ iff Γ and $\mathsf{exec}_\theta(I)$ is satisfiable. And indeed, this notion of coherence requires that intentions are internally consistent, as well as that the agent does not believe (too strongly) that the preconditions of the intended actions do not hold at the intended time, and after adding $\mathsf{exec}_\theta(I)$ to the agent's beliefs, the agent must also believe that the intended actions' postconditions hold (with a certain probability).

This example shows why it is important that we use a logic in which we can explicitly reason about the execution of actions, as well as their pre- and postconditions. And furthermore, *the example shows how restricting to bounded policies and finite paths still allows us to express what we wish, while simultaneously allowing for good computational properties, which we will see in the following sections.*

3 Model Checking

In this section, we state the model checking problem for PLBP, show that it is decidable, and analyze its complexity. Note that we restrict our attention to state formulas, as this suffices for our intended applications. The model checking problem for PLBP can be formulated as follows:

Instance An MDP $\mathbb{M}$ over $\mathscr{A}$ with associated pre_a and Post_a for all $a \in \mathscr{A}$, a state s in $\mathbb{M}$, and a PLBP formula φ, where the numbers are written in unary.[3]

Question Does it hold that $\mathbb{M}, s \models \varphi$?

Theorem 1. *The model checking problem for PLBP is decidable.*

Proof. We give an algorithm for solving the model checking problem. Consider the set of all state subformulas of φ, $Subf(\varphi)$ (including those occurring inside its path subformulas). Order $Subf(\varphi)$ in the order of increasing complexity, starting with propositions. Then label the states in $\mathbb{M}$ with formulas in $Subf(\varphi)$ that they satisfy, in order. The cases of propositions and Boolean connectives are trivial. Let us consider the case of $\Diamond^n_{\bowtie r}\Phi$. In order to determine whether to label some state s with $\Diamond^n_{\bowtie r}\Phi$, we need to check whether there is some policy π^s such that property $\mu^s_\pi(\{\mathsf{s} \in \mathsf{Paths}(\pi^s) \mid \mathsf{s} \models \Phi\}) \bowtie r$. A straightforward approach is to iterate over all possible n-step policies π^s . There are $O(|\mathscr{A}|^{|S^n|})$ of those, and each policy is of size $O(|\mathscr{A}| \times |S^n|)$. We can generate them one by one and represent them using space polynomial in the MDP and exponential in the formula. For each policy π^s, generate $\mathsf{Paths}(\pi^s)$. That set is of size $O((|S| \times |\mathscr{A}|)^n)$. Given that states on each path are already labeled with the state subformulas of Φ, it is easy (i.e. linear time) to evaluate Φ on all of them. If for a given policy π it holds that $\mu^s_\pi(\{\mathsf{s} \in \mathsf{Paths}(\pi^s) \mid \mathsf{s} \models \Phi\}) \bowtie r$, then label s with $\Diamond^n_{\bowtie r}\Phi$. Otherwise, if no policy satisfies this, s does not satisfy $\Diamond^n_{\bowtie r}\Phi$. □

The algorithm above gives a decision procedure in EXPSPACE. It has the advantage that it can return a policy satisfying $\Diamond^n_{\bowtie r}\Phi$ if it exists (and since policy size is exponential in the formula, we cannot do better). However, if a witness policy is not required, it is possible to do better.

Below we give a nondeterministic algorithm which, given a structure $M = (S, P, V)$ over $\mathscr{A}$ and a formula φ_0, returns the set of states $[\varphi_0]_M$ satisfying φ_0: $[\varphi_0]_M = \{s \mid M, s \models \varphi_0\}$ (see Algorithm 1). This algorithm uses only polynomial space.

Given φ_0, we produce a set of subformulas $Subf(\varphi_0)$ of φ_0 in the usual way. We then proceed by cases. For all formulas in $Subf(\varphi_0)$ apart from $\Diamond^n_{\bowtie r}\Phi$ the cases are standard. Labeling states with $\Diamond^n_{\bowtie r}\Phi$ makes use of a function MEASURE

[3]If n is written in binary, then e.g. a linear pass over a sequence of n states takes time exponential in the size of the formula, which we find counterintuitive.

Algorithm 1 Labelling φ_0

```
function PLBP-LABEL($M, \varphi_0$)
    for $\varphi' \in Subf(\varphi_0)$ do
        case $\varphi' = p$
            $[\varphi']_M \leftarrow V(p)$
        case $\varphi' = \neg\varphi$
            $[\varphi']_M \leftarrow S \setminus [\varphi]_M$
        case $\varphi' = \varphi \wedge \psi$
            $[\varphi']_M \leftarrow [\varphi]_M \cap [\psi]_M$
        case $\varphi' = \Diamond^n_{\bowtie r}\Phi$
            $[\varphi']_M \leftarrow \{ s \mid \text{MEASURE}(node_0(s), \Phi, n, 0) \bowtie r\}$
    return $[\varphi_0]_M$
```

Algorithm 2 Computing the measure of paths

```
function MEASURE($q, \Phi, n, P_\Phi$)
    if $d(q) > n$ then
        if $p(q) \models \Phi$ then $P_\Phi \leftarrow P_\Phi + \mu(p(q))$
        return $P_\Phi$
    else
        guess $a \in actions(s(q))$
        for $o \in outcomes(s(q), a)$ do
            $P_\Phi \leftarrow \text{MEASURE}(node(q, a, o), \Phi, n, P_\Phi)$
        return $P'_\Phi$
```

which returns the measure of paths generated by a policy (see Algorithm 2).

The function MEASURE takes four arguments: the current node q, the formula Φ, the depth of the policy n, and the measure of the paths generated by the policy so far which satisfy Φ. The algorithm proceeds by recursive depth-first search. Each branch in the search tree is represented by a sequence of nodes. A *node* is a structure that consists of a state of M and a finite path of nodes leading to this node from the root node. For each node q, we have a function $s(q)$ which returns its state, $a(q)$ which returns the action taken to reach $s(q)$, $p(q)$ which returns the nodes on the path to q and $d(q)$ which returns the length of the path to q. The function $node_0(s)$ returns the root node, i.e., a node q_0 such that $s(q_0) = s$ $a(q) = nil$, $p(q_0) = [\,]$ and $d(q_0) = 0$. The function $node(q, a, s')$ returns a node q' where $s(q') = s'$, $a(q') = a$, $p(q') = [p(q) \cdot q']$ and $d(q') = d(q) + 1$.

The base case is when the length of the branch, $d(q)$ is greater than n. If Φ is true on the path represented by this branch, we update P_Φ, the measure of the paths generated so far, by adding the probability of the current path to it, before returning; otherwise, we return the value of P_Φ passed as an argument. The recursive case is when the length of the branch is less than or equal to n. We extend the policy by guessing an action possible in $s(q)$ (this is the only non-deterministic step in the algorithm). For each possible outcome state o of executing action a in state $s(q)$, we call MEASURE to update P_Φ with the measure of paths through state o. When all the outcome states of a have been considered, we return P_Φ as the measure of the paths with prefix $p(n)$ where action a is chosen in $s(n)$. Note that execution of MEASURE requires only polynomial space. The recursion is bounded by n, and for each recursive call, we need to remember only the remaining outcomes of the chosen action yet to be considered, which is bounded by $|S|$. Keeping track of the measure of the paths generated by the policy so far which satisfy Φ requires constant space. MEASURE is a nondeterministic algorithm, so the complete model checking algorithm runs in nondeterministic polynomial space. Since NPSPACE = PSPACE by Savitch's theorem [Savitch, 1970], there is a PSPACE algorithm for solving the model checking problem for PLBP. This gives us inclusion in PSPACE. Below we also prove PSPACE-hardness.

Theorem 2. *The model checking problem for PLBP is PSPACE-complete.*

Proof. The hardness proof is by reduction of the QSAT (Satisfiability of Quantified Boolean Formulas) problem (which is a PSPACE-complete problem) to the model checking problem for PLBP. We use a reduction from QSAT inspired by [Bulling and Jamroga, 2010]. However, instead of the strategic setting of [Bulling and Jamroga, 2010], where a verifier has a strategy to evaluate the formula to true (enforce the 'yes' state) if and only if the formula is satisfiable, we work in the probabilistic setting, where an agent has a policy to reach the 'yes' state with probability 1 if and only if the QBF formula is satisfiable.

The details of the proof are given in the supplementary material. □

4 Satisfiabilty

The satisfiability problem for PLBP is as follows:

Instance A PLBP formula φ over $\mathscr{A}$ with associated pre_a and Post_a for all $a \in \mathscr{A}$. We assume that the numbers are written in unary.

Question Is there an MDP $\mathbb{M}$ over $\mathscr{A}$, pre_a, and Post_a, such that there exists a state s in $\mathbb{M}$ with $\mathbb{M}, s \models \varphi$?

We will now begin proving that the satisfiability problem is also decidable.

Denote by $\mathsf{pd}(\varphi)$ the *policy depth* of a formula φ, defined as $\mathsf{pd}(\varphi) = 1$ if φ is atomic, $\mathsf{pd}(\varphi \wedge \psi) = \max\{\mathsf{pd}(\varphi), \mathsf{pd}(\psi)\}$, $\mathsf{pd}(\neg\varphi) = \mathsf{pd}(\varphi)$, and $\mathsf{pd}(\Diamond^n_{\bowtie r}\Phi) = n + \max\{\mathsf{pd}(\varphi) \mid \varphi \text{ appears in } \Phi\}$. We then have the following.

Theorem 3 (Small model property). *If a formula φ is satisfiable, then it is also satisfiable in an MDP with at most as many states as a tree with depth $\mathsf{pd}(\varphi)$ and branching factor $\sum_{a\in\mathscr{A}} |\mathsf{Post}_a|$.*

Proof sketch. As the proof is a conceptually clear but technically involved extension of the method of unraveling, well-known in modal logic, we give details only in the supplementary material. The intuition is that we can unfold the MDP into a tree-shaped structure, branching for every executable action and possible outcome (i.e. each executable $a \in \mathscr{A}$ and $\mathsf{post}_{a,i} \in \mathsf{Post}_a$). This tree satisfies exactly the same formulas as the original MDP. Evaluating φ only requires us to search through the tree up to depth $\mathsf{pd}(\varphi)$. □

Theorem 4 (Satisfiability). *The satisfiability problem for PLBP is decidable in 2EXPSPACE.*

Proof sketch. We first describe a decision procedure and comment on its complexity afterwards. Using the bound y on the state count from Theorem 3, we iterate over sets $S = \{s_1, \dots, s_l\}$ of size at most y, and over valuations V on S considering only the variables appearing in φ (plus all pre- and postconditions). We require that for all $s \in S$ there is some a such that $s \models \mathsf{pre}_a$. We now show that we can define some P such that $\langle S, P, V\rangle, s_1 \models \varphi$, iff some first-order logic (FOL) sentence α_φ holds in the theory of real closed fields (RCF). We refer to [Chang and Keisler, 2012] for those unfamiliar with RCF. Using the well-known decidability of RCF, we then get a decision procedure.

We will encode P through variables $p_{s,t,a}$ for $s, t \in S$ and $a \in \mathscr{A}$, denoting $P_{s,a}(t)$ (if that is defined). Writing $\mathbf{p}$ for the sequence of all these variables, we consider a first-order formula $\beta(\mathbf{p})$ that is true in RCF iff the P encoded by $\mathbf{p}$ is well-defined (following the conditions of Definition 1).

Next, we consider for every $1 \leqslant n \leqslant \mathsf{pd}(\varphi)$, state s, n-step policy π^s, $X \subseteq \mathsf{Paths}(\pi^s)$, $\bowtie$ and $r \in \mathbb{Q} \cap [0, 1]$, a FOL formula $\gamma_{s,X,\pi,\bowtie,r}(\mathbf{p})$ such that if $\beta(\mathbf{p})$ holds in RCF, then $\gamma_{s,X,\pi,\bowtie,r}(\mathbf{p})$ holds iff $\mu^s_\pi(X) \bowtie r$ in the MDP encoded by $\mathbf{p}$. Note that $\mu^s_\pi(X) \bowtie r$ is an inequality with a sum of products, expressible in RCF.

We now simultaneously define FOL formulas $\delta_{\psi,s}$, $\kappa_{\Phi,\mathbf{h}}$, and $\lambda_{\Phi,s,n,\pi,X}$, respectively expressing "$s \models \psi$", "$\mathbf{h} \models \Phi$", and "X is the set of all $\mathbf{h} \in \mathsf{Paths}(\pi^s)$ such that $\mathbf{h} \models \Phi$".

The definitions of these are simple, and almost directly follow the semantics of our logic, with e.g. (i) $\delta_{\Diamond^n_{\bowtie r}\Phi,s} = \bigvee_{n\text{-step } \pi^s} \bigvee_{X\subseteq\mathsf{Paths}(\pi^s)} \lambda_{\Phi,s,n,\pi,X} \to \gamma_{s,X,\pi,\bowtie,r}$, (ii) $\kappa_{\mathsf{X}\Phi,\mathbf{h}} = \kappa_{\Phi,\mathbf{h}_\mathsf{X}}$, and (iii) $\lambda_{\Phi,i,n,\pi,X} = \bigwedge_{\mathbf{h}\in X} \kappa_{\Phi,\mathbf{h}} \wedge \bigwedge_{\mathbf{h}\in\mathsf{Paths}(\pi^s)\setminus X} \neg\kappa_{\Phi,\mathbf{h}}$.

Finally, we put $\alpha_\varphi = \exists\mathbf{p}\big(\beta(\mathbf{p}) \wedge \delta_{\varphi,s_1}(\mathbf{p})\big)$, finishing our description of the decision procedure.

Iterating over S and V proceeds for a double exponential number of iterations w.r.t. the input, by the bound given in Theorem 3. The FOL formula we construct in each iteration is also of double exponential size, as the $\delta_{\Diamond^n_{\bowtie r}\Phi,s}$ formula contains a disjunct for every policy and set of paths. Noting that α_φ is an existential formula, it follows from existential RCF being in PSPACE [Canny, 1988] that the overall procedure runs using double exponential space. □

5 Related Work

There are many logics designed to reason about time and probability. Most of these consider *infinite* traces/histories, in contrast to PLBP. Though we will state for each logic how the expressive power of that logic differs from that of PLBP, we also refer the interested reader to [Artale *et al.*, 2019] for a more general comparison of finite- and infinite-trace logics.

The logics PCTL [Hansson and Jonsson, 1989; Hansson and Jonsson, 1994] and pCTL* [Aziz *et al.*, 1995] extend the branching-time temporal logics CTL and CTL*, respectively, with operators $\mathbb{P}_{\bowtie r}$ such that $\mathbb{P}_{\bowtie r}\varphi$ is true at a state iff infinite paths starting from that state satisfy φ with probability $\bowtie r$. The logics have decidable model checking, polynomial-time for PCTL and polynomial-space for pCTL*, but the matter of whether satisfiability is decidable has been a long-standing open problem, as is the case for all the other logics we mention here. Formulas of PCTL and pCTL* are interpreted over discrete-time Markov chains, which are our MDPs without actions. PLBP is *not* a fragment or a restriction of pCTL* to finite traces. One obvious reason for this is the presence of actions; formulas with do expressions in Example 5 cannot be expressed in pCTL*.

Closer in spirit are extensions introduced in [Bianco and de Alfaro, 1995]. Their models are Markov chains, in which at each state a next-state probability distribution is nondeterministically selected, according to which the structure transitions to a next state. This is in effect working with a less structured notion of actions (without pre- and postconditions), in which the nondeterministic choice corresponds to a choice of action. The logic has operators $\mathbb{P}_{\geqslant r}$ and $\mathbb{P}_{\leqslant r}$, which respectively consider minimal and maximal probabilities of paths with respect to the nondeterministic choices. Considering the mentioned correspondence between nondeterminism and actions, these operators are effectively doing existential and universal quantification (respectively) over policies, similar to the $\Diamond$ and $\Box$ modalities of PLBP. The PCTL extension still has polynomial-time model checking, but the pCTL* extension now has a lower bound of 2EXPTIME. Though these extensions have some notion of action hiding behind the scenes as we pointed out, they cannot express everything we need for our applications (which we can express with PLBP). Specifically, we cannot reason in these extensions about the executability and consequences of certain actions, which is required to determine the coherence of an agent's intentions (as discussed in Example 6). Besides that, these extensions are quite limited in what kind of inequalities they support: for example, we cannot reason about the existence of a policy for which Φ holds with a probability less than 0.2.

The logics PATL and PATL*, introduced by [Chen and Lu, 2007], are extensions of PCTL and pCTL* respectively, in which there is a set of multiple agents, and in which actions are explicitly part of the semantics (but not the language). These logics are interpreted over probabilistic concurrent game structures, which are generalizations of MDPs in which the probability of transitioning from one state to another is determined by which actions all of the agents choose. The main modality of these logics is $\langle\langle B\rangle\rangle_{\bowtie r}$, where B is a subset of agents, with the interpretation that $\langle\langle B\rangle\rangle_{\bowtie r}\Phi$ iff the agents in B have a (memoryful) strategy/policy together that enforces that the path formula Φ holds with probability $\bowtie r$. Model checking PATL is in $P^{\mathrm{NP}\cap\mathrm{coNP}}$, and [Chen and Lu, 2007] state that pATL* also has decidable model checking, and though we did not manage to find the proof of this statement and (the complexity of) the corresponding algorithm, it must have a lower bound of 2EXPTIME since model checking for the pCTL* extension of [Bianco and de Alfaro, 1995] reduces to it. Again, PLBP is not a fragment or a restriction of these formalisms.

Probabilistic Strategy Logic (PSL) [Aminof *et al.*, 2019] has first-order quantification over memoryful strategies/policies, allowing one to express statements like 'there exists a policy such that p will hold with probability at least 0.2 and q will hold with probability precisely 0.9', which is not possible in PATL* (or any of the other logics mentioned here, including PLBP). We note that PSL does not allow us to reason about the execution of specific actions. This increase in expressive power comes with a high price: model checking is already undecidable for a highly restricted fragment of PSL, even when considering deterministic finite-memory strategies and a single agent. Limiting to memoryless strategies gives decidable model-checking with an upper bound of 3EXPSPACE. So even if we were to add reasoning about explicit pre- and postconditions into PSL, the resulting logic would not be practical for our purposes from a computational perspective, having too high complexity model checking and unknown decidability of satisfiability. And we truly need memoryful (bounded) policies for statements like "the agent can act in the next n steps in such a way that ... " which we use for checking the coherence of intentions.

6 Future Work

As explained in Section 1, the intended application of PLBP is to build a theory of intention revision under uncertainty, following [Shoham, 2009] in considering intentions as commitments towards time. In particular, we aim to follow [van Zee *et al.*, 2020] in putting intention revision on top of the well-known AGM-style belief revision [Alchourrón *et al.*, 1985]. The computational properties of PLBP are important for this goal; especially decidable satisfiability, as it allows us to compute whether beliefs and intentions after an update are still consistent. And as discussed in Example 6, we can define coherence as satisfiability in the logic. Furthermore, the logic working with finite/bounded traces and policies also facilitates AGM-style revision in the absence of compactness, following [van Zee *et al.*, 2020].

Besides intention revision, we plan to consider reasoning about policies in reinforcement learning. For this application, we will consider extensions of PLBP with reward signals and imperfect information. Natural starting points are [Jamroga, 2008], who introduces a quantitative logic for MDPs that reasons about rewards, and [Huang *et al.*, 2012], who introduce an extension of PATL* with imperfect information.

We will also investigate the expressive power of the logic, by following common practice in modal logic and defining an appropriate notion of bisimulation for MDPs that satisfies a Hennessy-Milner theorem with respect to PLBP. A sound and complete axiomatization of PLBP is of interest since it would allow us to formalize how pre- and postconditions behave in PLBP and verify whether principles of coherence like those listed by [Shoham, 2009] hold in the logic.

7 Conclusion

We have introduced PLBP, a novel probabilistic temporal logic for reasoning about the interactions between actions, time, and probability. It can describe time-bounded policies in MDPs, the probability that some property can be achieved by such a policy, and can also reason about specific properties of actions and policies. We have analyzed the typical reasoning problems for PLBP, namely model checking and satisfiability, and have shown that both are decidable. We plan to use PLBP in the future for formalizing intention revision under uncertainty and to apply it to settings like Reinforcement Learning.

References

[Alchourrón *et al.*, 1985] Carlos E. Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *The Journal of Symbolic Logic*, 50(2):510–530, 1985.

[Aminof *et al.*, 2019] Benjamin Aminof, Marta Kwiatkowska, Bastien Maubert, Aniello Murano, and Sasha Rubin. Probabilistic strategy logic. In Sarit Kraus, editor, *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI-19)*, pages 32–38, 2019.

[Artale *et al.*, 2019] Alessandro Artale, Andrea Mazzullo, and Ana Ozaki. Do you need infinite time? In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 1516–1522. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[Aziz *et al.*, 1995] Adnan Aziz, Vigyan Singhal, Felice Balarin, Robert K. Brayton, and Alberto L. Sangiovanni-Vincentelli. It usually works: The temporal logic of stochastic systems. In Pierre Wolper, editor, *Computer Aided Verification*, pages 155–165, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg.

[Bianco and de Alfaro, 1995] Andrea Bianco and Luca de Alfaro. Model checking of probabilistic and nondeterministic systems. In P. S. Thiagarajan, editor, *Foundations of Software Technology and Theoretical Computer Science*, pages 499–513, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg.

[Bulling and Jamroga, 2010] Nils Bulling and Wojciech Jamroga. Verifying agents with memory is harder than it seemed. *AI Commun.*, 23(4):389–403, 2010.

[Canny, 1988] John Canny. Some algebraic and geometric computations in PSPACE. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing STOC'88*, page 460–467. ACM, 1988.

[Chang and Keisler, 2012] C. C. Chang and H. Jerome Keisler. *Model Theory*. Dover Publications, 3 edition, 2012.

[Chen and Lu, 2007] Taolue Chen and Jian Lu. Probabilistic alternating-time temporal logic and model checking algorithm. In *Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007)*, volume 2, pages 35–39, 2007.

[Cohen and Levesque, 1990] PR Cohen and HJ Levesque. Intention is choice with commitment. *Artificial intelligence*, 1990.

[De Giacomo and Vardi, 2015] Giuseppe De Giacomo and Moshe Y. Vardi. Synthesis for LTL and LDL on finite traces. In Qiang Yang and Michael J. Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015*, pages 1558–1564. AAAI Press, 2015.

[Hansson and Jonsson, 1989] Hans Hansson and Bengt Jonsson. A framework for reasoning about time and reliability. In *[1989] Proceedings. Real-Time Systems Symposium*, pages 102–111, 1989.

[Hansson and Jonsson, 1994] Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5):512–535, 1994.

[Huang *et al.*, 2012] Xiaowei Huang, Kaile Su, and Chenyi Zhang. Probabilistic alternating-time temporal logic of incomplete information and synchronous perfect recall. In Jörg Hoffmann and Bart Selman, editors, *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*. AAAI Press, 2012.

[Icard *et al.*, 2010] Thomas Icard, Eric Pacuit, and Yoav Shoham. Joint revision of belief and intention. In *Proceedings of the Twelfth International Conference on Principles of Knowledge Representation and Reasoning*, KR'10, page 572–574. AAAI Press, 2010.

[Jamroga, 2008] Wojciech Jamroga. A temporal logic for markov chains. In Lin Padgham, David C. Parkes, Jörg P. Müller, and Simon Parsons, editors, *7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008), Volume 2*, pages 697–704. IFAAMAS, 2008.

[Puterman, 1994] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics. Wiley, 1994.

[Rao and Georgeff, 1991] Anand S. Rao and Michael P. Georgeff. Modeling rational agents within a BDI-architecture. In James F. Allen, Richard Fikes, and Erik Sandewall, editors, *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*, pages 473–484. Morgan Kaufmann, 1991.

[Savitch, 1970] Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970.

[Shoham, 2009] Yoav Shoham. Logical theories of intention and the database perspective. *Journal of Philosophical Logic*, 38(6):633–647, 2009.

[van der Hoek *et al.*, 2007] Wiebe van der Hoek, Wojciech Jamroga, and Michael Wooldridge. Towards a theory of intention revision. *Synthese*, 155(2):265–290, 2007.

[van Ditmarsch *et al.*, 2011] Hans van Ditmarsch, Tiago de Lima, and Emiliano Lorini. Intention change via local assignments. In Mehdi Dastani, Amal El Fallah Seghrouchni, Jomi Hübner, and João Leite, editors, *Languages, Methodologies, and Development Tools for Multi-Agent Systems*, pages 136–151, Berlin, Heidelberg, 2011. Springer.

[van Zee *et al.*, 2020] Marc van Zee, Dragan Doder, Leendert van der Torre, Mehdi Dastani, Thomas Icard, and Eric Pacuit. Intention as commitment toward time. *Artificial Intelligence*, 283, 2020.